Claims

Independent Claims

1. Computer-implemented method for real-time productivity guidance of a software developer operating a user-computing device, the method comprising:

   1.1. capturing, by a client-side screen-capture module executing on the user-computing device, pixel data that represents a current visual output of at least one display associated with the user-computing device;

   1.2. performing on-device analysis of the pixel data to produce (i) text extracted by optical-character-recognition and (ii) visual feature vectors that identify an active application window;

   1.3. determining a work-context, by fusing the extracted text and feature vectors with electronic task data that are (a) retrieved from at least one external project- or source-code-management system via an application-programming-interface and (b) specific to the developer;

   1.4. detecting, via at least one machine-learning model, a discrepancy between the work-context and an expected task specification contained in the electronic task data;

   1.5. generating, without human intervention and during an active work session, a feedback notification responsive to the discrepancy; and

   1.6. delivering the feedback notification in real time through at least one channel selected from: (i) an integrated-development-environment (IDE) plug-in, (ii) a system-tray agent, and (iii) a messaging-platform bot,

   wherein raw pixel data are discarded or redacted after the on-device analysis, and only derived context labels or aggregated metrics are transmitted beyond the user-computing device.

2. The method of claim 1, wherein the feedback notification comprises a suggestion to refocus on a higher-priority task when the detected discrepancy indicates activity on an unassigned code module.

3. The method of claim 1, wherein the machine-learning model dynamically adjusts a threshold for flagging idle time according to a historical behavior profile of the developer.

4. The method of claim 1, wherein the feedback notification escalates from a first channel to a second channel when a same discrepancy persists for a configurable period.

5. The method of claim 1, further comprising obfuscating personally identifiable or unrelated private information in the pixel data prior to the on-device analysis by applying selective blurring.

6. The method of claim 1, wherein the screen-capture module captures video frames at a first sampling rate and, upon detecting a rapid series of developer interactions, automatically increases the sampling rate.

7. The method of claim 1, further comprising storing, in a secure audit log, a timestamped record that the feedback notification was delivered and subsequently acknowledged by the developer.

8. The method of claim 1, wherein the electronic task data include a current sprint backlog retrieved from a project-management system and commit metadata retrieved from a version-control repository.

9. The method of claim 1, further comprising computing a composite developer-alignment score that combines (i) a focus-percentage metric indicating time spent on task-aligned windows and (ii) a context-switching frequency metric; and displaying the composite developer-alignment score to the developer.

10. The method of claim 1, wherein the machine-learning model is at least partially trained on anonymized historical datasets that map screen-derived contexts to known task outcomes, thereby enabling anomaly detection.

---

11. System for real-time productivity guidance of software developers, comprising:

11.1. at least one user agent configured to execute on a developer workstation and including:

- a screen-capture submodule;
- an on-device vision-analysis submodule; and
- a local privacy-filter submodule;

11.2. a context-analyzer server communicatively coupled to the user agent and configured to merge screen-derived context with external project-management data;

11.3. an artificial-intelligence feedback engine configured to (a) detect task misalignment in near-real-time and (b) emit feedback events; and

11.4. a multi-channel notification dispatcher configured to route the feedback events to at least one of an IDE plug-in, a messaging-platform bot, or a system-tray notification component,

wherein the user agent is further configured to transmit only context labels or aggregated metrics, and not raw screenshots, to the context-analyzer server.

12. The system of claim 11, wherein the IDE plug-in is operable to display focus suggestions in a non-modal panel that integrates with code-lint warnings.

13. The system of claim 11, wherein the messaging-platform bot is configured to receive a natural-language explanation from the developer and pass the explanation to the feedback engine for semantic analysis.

14. The system of claim 11, wherein the screen-capture submodule and the on-device vision-analysis submodule are implemented using a shared graphics-processing-unit pipeline to reduce latency.

15. The system of claim 11, wherein the artificial-intelligence feedback engine executes a fairness policy that suppresses penalty escalation when the context-analyzer server identifies a blocking external dependency for the current task.

16. The system of claim 11, wherein the privacy-filter submodule encrypts any temporary pixel buffers using a device-specific key stored in a trusted-platform-module.

17. The system of claim 11, wherein the context-analyzer server exposes a representational-state-transfer (REST) interface that enables third-party monitoring platforms to stream activity logs into the feedback engine.

---

18. Non-transitory computer-readable medium storing instructions that, when executed by one or more processors of a computing device, cause the computing device to perform the method of claim 1.

Dependent Claims (method or system, as apparent from context)

19. wherein the feedback notification includes positive reinforcement when a measured focus duration exceeds a developer-specific median by at least 20 percent.

20. wherein the discrepancy detection further incorporates a rule that flags opening of a website classified by a site-category model as “non-work” for more than a preset duration.

21. wherein the local privacy-filter submodule redacts pixel regions containing regular-expression matches to patterns indicative of personal e-mail addresses.

22. wherein aggregation of metrics is performed over rolling ten-minute windows to satisfy GDPR data-minimization principles.

23. wherein the system further comprises a dashboard that visualizes, per developer and per sprint, (i) cumulative alignment score, (ii) frequency of feedback events, and (iii) mean acknowledgment latency.

24. wherein the feedback engine triggers a manager-visible escalation event only after at least two unresolved discrepancies of a same type within a configurable working day.

25. wherein the user agent operates in a scheduled mode in which monitoring is automatically suspended outside of defined working hours to prevent non-work surveillance.

26. wherein the IDE plug-in injects hyperlinks to relevant knowledge-base articles when the developer appears stuck—determined by repetitive error logs coupled with frequent undo operations.

27. wherein the messaging-platform bot can update task status in the external project-management system when the developer responds with a pre-defined structured command.

28. wherein the artificial-intelligence feedback engine adapts over time by updating per-developer thresholds via online reinforcement learning using acknowledgment outcomes as reward signals.

29. wherein the system is deployable in a stand-alone vertical configuration in which the context-analyzer server, feedback engine, and notification dispatcher are hosted on-premises, isolated from external cloud services.

30. wherein the system is deployable in a plug-in configuration in which the screen-capture function is provided by a third-party monitoring platform and the user agent comprises only a lightweight API connector.

31. wherein the system is deployable in a hybrid modular configuration comprising (i) an IDE plug-in only, (ii) a system-tray agent only, or (iii) both, with interoperability achieved through a message-queue protocol.

2 • New Independent Claim (method)

32. Computer-implemented method for automated task-board management in conjunction with the method of claim 1, the method comprising:

    32.1. querying, via an application-programming-interface, a task-board service to determine whether a card corresponding to the work-context exists;

    32.2. when no such card exists, creating a new task card on the task-board service, the card comprising a title derived from the work-context and metadata that identify at least one of: a repository branch, a file path, or a commit identifier;

    32.3. updating the task card in real time by appending timestamped entries that reflect developer activities detected by the method of claim 1; and

    32.4. changing a status of the task card responsive to a completion condition determined by the machine-learning model, whereby manual task-board administration work by the developer is reduced.

---

3 • Representative Dependent Claims

33. The method of claim 32, wherein each timestamped entry further comprises an automatically redacted code diff that masks sensitive literals.

34. The method of claim 32, wherein the task card is automatically moved from an “In Progress” list to a “Done” list when (i) a pull request associated with the work-context is

merged, or (ii) all unit tests referenced in the work-context pass in a continuous-integration pipeline.

35. The method of claim 32, wherein natural-language comments appended to the task card are generated by a large-language-model component of the AI feedback engine and summarize one or more developer actions over a rolling ten-minute period.

36. The method of claim 32, wherein privacy-filtering is applied locally such that any personally identifiable information present in the developer's screen content is excluded from the task card.

37. The method of claim 32, wherein an escalation rule notifies a project manager when the task card remains in a "Blocked" column for longer than a configurable threshold without acknowledgment by the developer