SWED Übung 10

**1. Project Plan for a Tetris AR App**

The entire project is planned over approximately six months.

**Project Phases and Activities:**

* **Phase 1: Requirements & Concept Design (3 Weeks)**: This initial phase focuses on defining the core game mechanics, features, and the target AR platform (e.g., specific AR glasses). The team will establish the project scope and create a detailed specification document.
* **Phase 2: UI/UX Design (3 Weeks)**: Following the concept design, the design team will create the user interface (UI) and user experience (UX) tailored for an augmented reality environment. This includes designing menus, game overlays, and interaction methods suitable for AR glasses.
* **Phase 3: 3D Asset Creation (5 Weeks)**: Parallel to UI/UX design, 3D artists will begin creating the necessary visual assets. This includes the 3D models for the various Tetris blocks (Tetrominos), the game board (well), and any environmental or special effect elements.
* **Phase 4: Core Game Development (8 Weeks)**: With the UI/UX design complete, developers will program the fundamental game logic. This includes piece generation, movement, rotation, line clearing, and scoring.
* **Phase 5: AR Integration (8 Weeks)**: Once the core mechanics are functional, they will be integrated into the AR environment. This involves using an AR development kit (like ARKit or ARCore) to handle real-world surface detection, anchor the game in the user's space, and render the 3D assets correctly.
* **Phase 6: Testing & QA (4 Weeks)**: A dedicated quality assurance phase is crucial for testing all aspects of the game. This includes gameplay bug hunting, performance testing on different AR devices, and ensuring the user experience is seamless.
* **Phase 7: Deployment & Release (2 Weeks)**: The final phase involves preparing the app for launch, submitting it to the relevant app stores, and executing the marketing plan for the release.

**Dependencies and Milestones:**  
Project activities have clear dependencies; for instance, core game development cannot begin until the UI/UX design is approved, and AR integration depends on both the core game and the 3D assets being ready. Key milestones include the completion of the concept document, a functional prototype, the alpha build, the beta build, and the final release candidate.

Ein Bild, das Screenshot, Reihe, Diagramm enthält.

KI-generierte Inhalte können fehlerhaft sein.

Gantt chart for Tetris Gaming App (AR glasses) project plan with activities, dependencies, and milestones

Milestones:

1. 22.01.26 Concept approved
2. 26.02.26 Design & Assets complete
3. 09.04.26 Functional prototype
4. 04.06.26 Alpha version ready
5. 02.07.26 Beta version stable
6. 16.07.26 Project completed

**2. Cost Estimate and Pricing Strategy**

**Project Cost Estimation**  
The cost of developing an AR game varies significantly based on its complexity, features, and the development team's location. For a Tetris AR app with high-quality graphics and smooth AR integration, the cost would likely fall into the mid-range category.

* **Simple AR Games**: Can cost between **$50,000 and $80,000**.
* **Multi-featured AR Games**: With more complex features and higher-end UI/UX, costs range from **$85,000 to $120,000**.
* **Advanced AR Games**: Games using features like GPS and complex real-world interaction can exceed **$200,000**.

**Rationale for an estimated cost of $95,000:**  
A Tetris AR app is more complex than a basic AR application but doesn't require advanced features like location-based services. The estimate accounts for:

* **Team Salaries**: A team of developers, a UI/UX designer, a 3D artist, and a project manager over a 6-month period.
* **Technology Stack**: Licensing for a game engine like Unity or Unreal Engine and costs associated with AR SDKs and 3D modeling software.
* **Feature Implementation**: Costs for developing core game logic and specific AR functionalities like gesture control ($20,000–$50,000) and 3D modeling/rendering ($5,000–$30,000).

**Software Pricing Strategy**  
The recommended pricing model would be **Freemium with In-App Purchases (IAPs)**.

**Explanation:**

* **Low Barrier to Entry**: Offering the game for free would encourage a maximum number of downloads and allow users to experience the novelty of AR Tetris without an initial investment.
* **Revenue Generation**: Revenue would be generated from optional IAPs. To preserve the classic, skill-based nature of Tetris, these purchases should be purely cosmetic. Examples include custom-designed Tetromino skins, different visual themes for the game board, or unique background effects.
* **User Retention**: This model provides ongoing value to engaged players who wish to personalize their experience, which can lead to higher long-term revenue compared to a one-time purchase.

**3. Staffing for Minimal Time-to-Market**

To achieve the minimal time to market, the project should be staffed with a skilled, well-coordinated team and adopt efficient development strategies.

* **Adopt an MVP Approach**: Focus the initial development effort on creating a Minimum Viable Product (MVP). This version would include only the essential features needed to deliver the core Tetris AR experience. Additional features can be added in subsequent updates. This approach accelerates the initial launch and allows for gathering crucial user feedback early.
* **Assemble a Skilled Core Team**: The team should consist of specialists with experience in game development and AR technologies.
  + **Project Manager**: To ensure smooth workflow and clear communication.
  + **Lead AR Developer**: An expert in Unity/Unreal and ARKit/ARCore to guide the technical implementation.
  + **2-3 Game Developers**: To work on different components in parallel.
  + **1 UI/UX Designer**: Focused on creating an intuitive AR interface.
  + **1-2 3D Artists**: To produce high-quality assets efficiently.
  + **2 QA Testers**: Integrated from the start to provide continuous testing.
* **Use Staff Augmentation**: To accelerate development, consider hiring external experts or contractors for specialized tasks or to bridge temporary manpower gaps. This provides quick access to skills without a long-term hiring commitment.
* **Encourage Parallel Workflows**: Structure the project plan (as seen in the Gantt chart) to allow tasks to be completed in parallel. For example, 3D asset creation can occur at the same time as UI/UX design.
* **Promote Efficient Cooperation**: A key to speed is eliminating process waste, such as over-engineering features or waiting for decisions. An effective cooperation strategy with clear communication channels is essential to prevent bottlenecks.

**4. Software Development Process**

The most suitable software development process for this project is **Agile, specifically using the Scrum framework**.

**Explanation:**

* **Flexibility and Adaptation**: AR is a relatively new field, and game development often involves discovering what is fun and functional through experimentation. The Agile process is designed to accommodate change. The project can be adjusted after each development cycle (sprint) based on new insights or technical challenges.
* **Iterative Development**: The project is broken down into short, iterative cycles called sprints (e.g., 2 weeks). At the end of each sprint, the team delivers a functional piece of the product. This allows for early and frequent testing of the AR experience on actual devices, which is critical for tuning gameplay and performance.
* **Early Feedback**: The iterative nature of Agile allows for gathering feedback from stakeholders and test users throughout the development process, not just at the end. This helps ensure the final product meets user expectations.
* **Focus on MVP**: Agile methodologies align perfectly with an MVP strategy. The team can prioritize the most critical features for the initial release and then build upon the product in subsequent sprints, reducing the initial time to market.

**5. Options for an Over-Budget and Over-Time Project**

When a project is over budget and behind schedule, several corrective actions can be taken. Here are five options:

1. **Reduce Project Scope**: This is often the most impactful solution. Work with stakeholders to identify and remove non-essential features from the current project. This reduces the amount of work remaining, saving both time and money. The deferred features can be planned for a future update after the core product is successfully launched.
2. **Fast-Track the Schedule**: Examine the remaining tasks in the project plan and identify any that are scheduled sequentially but could be performed in parallel (or overlapped). For instance, certain QA testing activities could begin on completed modules while development continues on others. This technique saves time without adding cost, but it can increase risk and management complexity.
3. **"Crash" the Schedule with More Resources**: If the deadline is immovable, you can add more resources to the project to accelerate the remaining work. This could mean hiring temporary contractors, authorizing overtime, or bringing in more experienced (and thus more productive) developers. This option will increase the project's overall cost but can help meet a critical deadline.
4. **Enforce Strict Scope Control**: Implement a "zero tolerance" policy for any scope changes. Prevent any new requests or "under the radar" additions from being worked on. The team's entire focus must be on completing the already-defined scope of work. This ensures that all remaining effort and budget are directed toward finishing the project as originally planned.
5. **Renegotiate Deadline and Budget**: Present a transparent report to the project sponsors and stakeholders. Explain the reasons for the overruns, the current status, and the forecast for completion. Based on this, formally request an extension of the deadline and/or an increase in the budget. This should be accompanied by a revised, realistic plan for completing the project with the additional resources.