DwellinglyAI MVP Development Project

Comprehensive Overview: The DwellinglyAI MVP development project aims to design and develop a Minimum Viable Product (MVP) for a B2C web application, leveraging Microsoft technologies to demonstrate core functionalities and attract investors for full-scale development. The project will focus on developing a scalable and secure platform, implementing essential features, integrating AI-driven valuation tools, and creating a user-friendly interface.

Logical Structure:

Requirements Gathering

System Design

Setup Development Environment

Backend Development

Frontend Development

Integration

Basic Testing

Demo Preparation

Key Deliverables:

Requirements Document (after Requirements Gathering phase)

System Design Document (after System Design phase)

Development Environment Setup (after Setup Development Environment phase)

Backend API Documentation (after Backend Development phase)

Frontend UI Prototype (after Frontend Development phase)

Integrated MVP (after Integration phase)

Test Report (after Basic Testing phase)

Demo Script and Presentation (after Demo Preparation phase)

As the Solution Architect, I will provide a detailed high-level system architecture for the DwellinglyAI MVP, ensuring scalability, security, and integration with Microsoft technologies.

System Architecture:

The proposed system architecture for NexHomeAgent MVP consists of the following components:

Presentation Layer:

Blazor WebAssembly (WASM) for the frontend, providing a fast and secure user interface.

Hosted on Azure App Service, ensuring scalability and high availability.

Application Layer:

ASP.NET Core Web API for the backend, handling business logic and data processing.

Hosted on Azure App Service, ensuring scalability and high availability.

Data Access Layer:

Azure SQL Database for storing property listings, user data, and other relevant information.

Utilizing Entity Framework Core for data modeling and database schema management.

AI/ML Layer:

Azure Machine Learning for developing and deploying AI-driven valuation tools and personalized property recommendations.

Integrating with the backend API for seamless data exchange.

Security and Authentication:

Azure Active Directory (AAD) for user authentication and authorization.

Implementing OAuth 2.0 and OpenID Connect for secure authentication and authorization.

Integration and APIs:

Azure API Management for managing and securing backend APIs.

Utilizing Azure Service Bus for message queuing and integration with other services.

System Components:

Property Search Service:

Responsible for handling property search queries and returning relevant results.

Utilizing Azure Search for efficient and scalable search functionality.

Property Details Service:

Responsible for retrieving and displaying detailed property information.

Integrating with Azure SQL Database for data retrieval.

Favorites Service:

Responsible for managing user favorites and providing personalized recommendations.

Utilizing Azure Cosmos DB for storing user preferences and favorites.

CMA Service:

Responsible for generating Comparative Market Analysis (CMA) reports.

Integrating with Azure Machine Learning for AI-driven valuation tools.

Scalability and Security:

Auto-Scaling:

Configuring Azure App Service to auto-scale based on traffic and resource utilization.

Load Balancing:

Utilizing Azure Load Balancer for distributing traffic across multiple instances.

Security Groups:

Configuring Azure Network Security Groups (NSGs) for controlling inbound and outbound traffic.

Encryption:

Implementing encryption for data at rest and in transit using Azure Key Vault and SSL/TLS certificates.

Next Steps:

Review and finalize the system architecture with the project team.

Begin setting up the development environment, including Azure Repos and Visual Studio.

Start implementing the backend API using ASP.NET Core and Azure SQL Database.

Develop the frontend UI using Blazor and integrate with the backend API.

Please let me know if you have any questions or concerns regarding the proposed system architecture.

backend\_developer:

As the backend developer, I will focus on implementing the core backend functionalities, developing data models, and creating API endpoints using ASP.NET Core and Azure SQL Database.

Backend Development Plan:

Based on the system architecture provided by the solution architect, I will implement the following backend components:

Property Search Service:

Create a RESTful API endpoint for handling property search queries.

Utilize Entity Framework Core to interact with the Azure SQL Database and retrieve relevant property listings.

Implement filtering and sorting functionality using LINQ.

Property Details Service:

Create a RESTful API endpoint for retrieving detailed property information.

Use Entity Framework Core to retrieve data from the Azure SQL Database.

Implement caching using Redis to improve performance.

Favorites Service:

Create a RESTful API endpoint for managing user favorites.

Use Entity Framework Core to interact with the Azure SQL Database and store user preferences.

Implement a caching mechanism using Redis to improve performance.

CMA Service:

Create a RESTful API endpoint for generating Comparative Market Analysis (CMA) reports.

Integrate with Azure Machine Learning to leverage AI-driven valuation tools.

Use Entity Framework Core to retrieve data from the Azure SQL Database.

Data Modeling:

I will design the database schema using Entity Framework Core, focusing on the following entities:

Properties:

Property ID (primary key)

Address

City

State

Zip

Price

Number of bedrooms

Number of bathrooms

Square footage

Users:

User ID (primary key)

Username

Email

Password (hashed)

Favorites:

Favorite ID (primary key)

User ID (foreign key)

Property ID (foreign key)

API Endpoints:

I will create the following API endpoints:

GET /properties: Retrieve a list of properties based on search criteria.

GET /properties/{id}: Retrieve detailed information about a specific property.

POST /favorites: Add a property to a user's favorites list.

GET /favorites: Retrieve a list of properties in a user's favorites list.

GET /cma: Generate a Comparative Market Analysis (CMA) report for a specific property.

Next Steps:

Set up the development environment, including Visual Studio and Azure Repos.

Implement the backend API using ASP.NET Core and Azure SQL Database.

Integrate with Azure Machine Learning for AI-driven valuation tools.

Conduct unit testing and integration testing to ensure the backend API is functioning correctly.

Please let me know if you have any questions or concerns regarding the backend development plan.

frontend\_developer:

As the frontend developer, I will focus on implementing the core frontend components, developing the user interface for property search and listings, and integrating with the backend APIs using Blazor.

Frontend Development Plan:

Based on the system architecture provided by the solution architect, I will implement the following frontend components:

Property Search Component:

Create a user-friendly search interface using Blazor, allowing users to input search criteria such as location, price range, and number of bedrooms.

Use the backend API to retrieve a list of properties based on the search criteria.

Implement filtering and sorting functionality using JavaScript and CSS.

Property Details Component:

Create a detailed property view using Blazor, displaying comprehensive property information, including photos, descriptions, and pricing information.

Use the backend API to retrieve detailed property information.

Implement a caching mechanism using LocalStorage to improve performance.

Favorites Component:

Create a favorites list view using Blazor, allowing users to view and manage their saved properties.

Use the backend API to retrieve a list of properties in the user's favorites list.

Implement a caching mechanism using LocalStorage to improve performance.

CMA Component:

Create a Comparative Market Analysis (CMA) report view using Blazor, displaying estimated property values and market insights.

Use the backend API to generate a CMA report for a specific property.

UI Design:

I will design the user interface using a combination of HTML, CSS, and JavaScript, focusing on a responsive and intuitive design. The UI will include the following features:

Home Page:

A search bar with autocomplete functionality

A list of featured properties

A call-to-action to encourage users to search for properties

Property Search Results:

A list of properties matching the search criteria

Filtering and sorting options

A map view to display property locations

Property Details:

A detailed property view with photos, descriptions, and pricing information

A favorites button to allow users to save properties

Favorites List:

A list of saved properties

A remove button to allow users to remove properties from their favorites list

Blazor Implementation:

I will use Blazor to implement the frontend components, taking advantage of its features such as:

Components: I will create reusable UI components using Blazor, making it easy to maintain and update the UI.

Data Binding: I will use Blazor's data binding features to bind data from the backend API to the UI components.

Event Handling: I will use Blazor's event handling features to handle user interactions, such as clicking on a property to view its details.

Next Steps:

Set up the development environment, including Visual Studio and Azure Repos.

Implement the frontend components using Blazor.

Integrate with the backend API using RESTful API endpoints.

Conduct unit testing and integration testing to ensure the frontend is functioning correctly.

Please let me know if you have any questions or concerns regarding the frontend development plan.

aiml\_engineer:

As the AI/ML engineer, I will focus on implementing the AI-driven valuation tools, developing personalized property recommendations, and integrating chatbot functionality using Azure Machine Learning, AI algorithms, and NLP.

AI/ML Development Plan:

Based on the system architecture provided by the solution architect, I will implement the following AI/ML components:

Property Valuation and CMA:

Use Azure Machine Learning to train a machine learning model that estimates property values based on historical data and market trends.

Integrate the trained model with the backend API to generate CMA reports for specific properties.

Personalized Property Recommendations:

Use AI algorithms to analyze user behavior, search history, and saved properties to suggest personalized property recommendations.

Integrate the recommendation engine with the backend API to provide users with relevant property suggestions.

Chatbot Integration:

Use NLP to develop a chatbot that can understand user queries and provide relevant responses.

Integrate the chatbot with the backend API to provide users with a conversational interface for searching properties and getting recommendations.

Azure Machine Learning Implementation:

I will use Azure Machine Learning to train and deploy the machine learning models for property valuation and CMA. The implementation will include:

Data Preparation:

Collect and preprocess historical property data, including sales prices, property features, and market trends.

Split the data into training and testing sets.

Model Training:

Train a machine learning model using the training data, selecting the best algorithm and hyperparameters based on model performance.

Evaluate the model's performance using metrics such as mean absolute error and R-squared.

Model Deployment:

Deploy the trained model to Azure Machine Learning, creating a RESTful API endpoint for generating CMA reports.

AI Algorithm Implementation:

I will use AI algorithms to develop the personalized property recommendation engine. The implementation will include:

Data Collection:

Collect user behavior data, including search history and saved properties.

Preprocess the data to extract relevant features.

Model Training:

Train a recommendation model using the collected data, selecting the best algorithm and hyperparameters based on model performance.

Evaluate the model's performance using metrics such as precision and recall.

Model Deployment:

Deploy the trained model to the backend API, creating a RESTful API endpoint for generating personalized property recommendations.

NLP Implementation:

I will use NLP to develop the chatbot, integrating it with the backend API to provide users with a conversational interface. The implementation will include:

Intent Identification:

Use NLP to identify user intents, such as searching for properties or getting recommendations.

Map the identified intents to relevant API endpoints.

Entity Extraction:

Use NLP to extract relevant entities from user queries, such as location or property type.

Use the extracted entities to generate API requests.

Response Generation:

Use NLP to generate human-like responses to user queries, based on the API responses.

Next Steps:

Set up the Azure Machine Learning environment, including creating a workspace and setting up data storage.

Implement the property valuation and CMA model using Azure Machine Learning.

Implement the personalized property recommendation engine using AI algorithms.

Integrate the chatbot with the backend API using NLP.

Please let me know if you have any questions or concerns regarding the AI/ML development plan.

quality\_assurance\_qa\_engineer:

As the Quality Assurance (QA) Engineer, I will ensure that the MVP meets the requirements and works as intended. Here's my plan to conduct basic functionality tests, identify and report defects, and ensure the MVP meets the requirements:

Testing Scope:

Primary Functionalities:

Property Search

Detailed Property Views

Comparative Market Analysis (CMA)

Saving Favorites

Personalized Property Recommendations

Development Phases:

Backend Development

Frontend Development

Integration

Hosting Setup

User Stories:

As a buyer, I want to search for properties by specifying criteria such as location, price range, and number of bedrooms.

As a buyer, I want to view detailed information about a property, including photos, descriptions, and pricing information.

As a buyer, I want to save properties to a favorites list for future reference.

Testing Approach:

Manual Testing: Conduct manual testing to ensure the MVP meets the requirements and works as intended.

Test Cases: Create test cases based on the user stories, primary functionalities, and development phases.

Test Data: Prepare test data to cover different scenarios and edge cases.

Defect Reporting: Report defects and track them using Azure DevOps.

Test Cases:

Property Search:

Test search functionality with different criteria (location, price range, number of bedrooms)

Test search results pagination and filtering

Test search results sorting

Detailed Property Views:

Test property details page with different properties

Test property photos and descriptions

Test pricing information

Comparative Market Analysis (CMA):

Test CMA report generation for different properties

Test CMA report content and accuracy

Saving Favorites:

Test saving properties to favorites list

Test viewing saved properties

Test removing properties from favorites list

Personalized Property Recommendations:

Test personalized property recommendations based on user behavior and preferences

Test recommendation accuracy and relevance

Test Environment:

Development Environment: Use the development environment set up by the development team.

Test Data: Use test data prepared by the development team.

Azure DevOps: Use Azure DevOps to track defects and test cases.

Timeline:

Test Planning: 2 days

Test Case Development: 4 days

Test Execution: 8 days

Defect Reporting and Tracking: Ongoing

Next Steps:

Review the test plan with the development team.

Develop test cases based on the user stories and primary functionalities.

Prepare test data to cover different scenarios and edge cases.

Execute manual testing and report defects using Azure DevOps.

Please let me know if you have any questions or concerns regarding the testing plan.