ŽILINSKÁ UNIVERZITA V ŽILINE
Fakulta riadenia
a informatiky

Fakulta riadenia a informatiky

Tester na maturitné testy

MATURANT

MÁRIO ŽILINČÍK

Študijný odbor: Informatika a riadenie
Školiace pracovisko: Žilinská Univerzita v Žiline
Vedúci: doc. Ing. Patrik Hrkút, PhD.
Žilina 2024

Obsah

ÚVOD

Pre svoju semestrálnu prácu z predmetu VAMZ som sa rozhodol vyvinúť aplikáciu určenú predovšetkým pre študentov maturitného ročníka na stredných školách. Táto aplikácia obsahuje staré maturitné testy, ako aj aktuálny maturitný test zo slovenského jazyka, pričom všetky testy sú interaktívne. Používatelia majú možnosť odpovedať na dva typy otázok – doplňovacie otázky a otázky s výberom odpovede, čo presne zodpovedá formátu skutočných maturitných testov.

Aplikácia je vybavená nastaviteľnou časomierou, ktorá umožňuje používateľom zvoliť si, koľko času potrebujú na vypracovanie testu – štandardne je to 100 minút, ako pri reálnych maturitných skúškach.

Hlavným cieľom aplikácie je simulovať reálne maturitné testy, preto nemôže chýbať ani funkcia vyhodnotenia testov. Používatelia môžu kedykoľvek vyhodnotiť svoje odpovede a aplikácia im vizuálne zobrazí, ktoré odpovede boli správne. Pri otázkach s výberom odpovede aplikácia navyše poskytne informáciu o správnej odpovedi. Okrem toho si používatelia môžu svoje výsledky uložiť.

Okrem toho aplikácia obsahuje aj rôzne základné témy z gramatiky a literatúry, ktoré môžu slúžiť pri príprave na samotné maturitné testy.

1 POPIS A ANALÝZA RIEŠENÉHO PROBLÉMU

1.1 ŠPECIFIKÁCIA ZADANIA, DEFINOVANIE PROBLÉMU

1.1.1 ŠPECIFIKÁCIA ZADANIA

Zadanie semestrálnej práce je relatívne jednoduché – vytvorenie aplikácie, ktorá pomáha študentom pripraviť sa na maturitné testy zo slovenského jazyka. Aplikácia bude obsahovať rôzne učebné materiály z gramatických tém, ako sú gramatické štýly a pravidlá, a literárnych tém, ako sú autori a ich diela. Aplikácia bude taktiež obsahovať testovací modul, ktorý má za úlohu simulovať reálne maturitné testy.

1.1.2 DEFINOVANIE PROBLÉMU

Hlavným cieľom aplikácie je poskytnúť študentom efektívny nástroj na prípravu na maturitné skúšky, pričom existuje viacero problémov, ktoré aplikácia rieši:

- **Nedostatok interaktívnych učebných nástrojov**

  Tradičné študijné materiály často postrádajú interaktivitu a okamžitú spätnú väzbu, čo môže byť pre študentov demotivujúce, pričom veľa študentov v dnešnej dobe uprednostňuje najmä učenie sa rôznymi digitálnymi štýlmi. Preto aplikácia ponúka interaktívne testy a okamžité vyhodnocovanie odpovedí, čím zvyšuje efektivitu učenia.

- **Zložité testovanie a príprava**

  Ako študent ktorý si taktiež prešiel prípravou na maturitný test zo slovenského jazyka si pamätám aj samotnú prípravu – tlač maturitných testov a následné ručné vypĺňanie či ručné vypisovanie odpovedí do textového súboru a následné ručné kontrolovanie správnosti odpovedí. Všetky tieto procesy boli veľmi zdĺhavé a zbytočne náročné, pričom mi zabrali veľa času. Aplikácia úplne eliminuje tieto zložité spôsoby testovania a prípravy, keďže majú používatelia všetky testy pokope a aplikácia sama vyhodnotí správnosť odpovedí.

- **Sledovanie pokroku**

  Študenti potrebujú možnosť sledovať svoj pokrok a porovnávať si svoje výsledky, čo aplikácia umožňuje pri možnosti uloženia výsledku testu.

1.2 PODOBNÉ APLIKÁCIE

1.2.1 Maturita vo vrecku

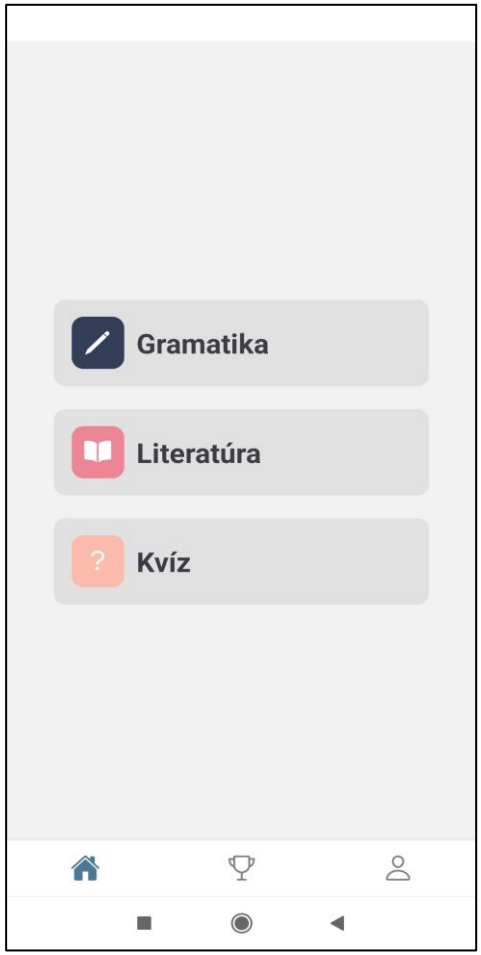
Aplikácia Maturita vo vrecku je podobnou aplikáciou, avšak má trochu odlišné zameranie – neobsahuje samotné maturitné testy, ale používateľom ponúka rôzne teoretické vedomosti zo slovenského jazyka a literatúry.

Funkcionality:

- Podrobné popísanie rôznych gramatických tém (Administratívny štýl, bibliografia, citoslovcia..)
- Podrobné popísanie literárnych diel od rôznych autorov (Andrej Sládkovič, Agatha Christie..)
- Implementácia kvízového systému, používateľ má možnosť si vybrať počet otázok (**Malý** – 5 otázok, **Stredný** – 10 otázok, **Veľký** – 15 otázok), okrem toho má možnosť si vybrať zameranie otázok (Literatúra, Gramatika, či zmiešané otázky oboch)
- Registračný systém – spoločne s možnosťou úpravy profilu (taktiež implementácia „avataru“ používateľa, avšak nie je dokončený)
- Rebríček študentov, ktorý obsahuje meno používateľa, počet jeho bodov a názov jeho strednej školy.

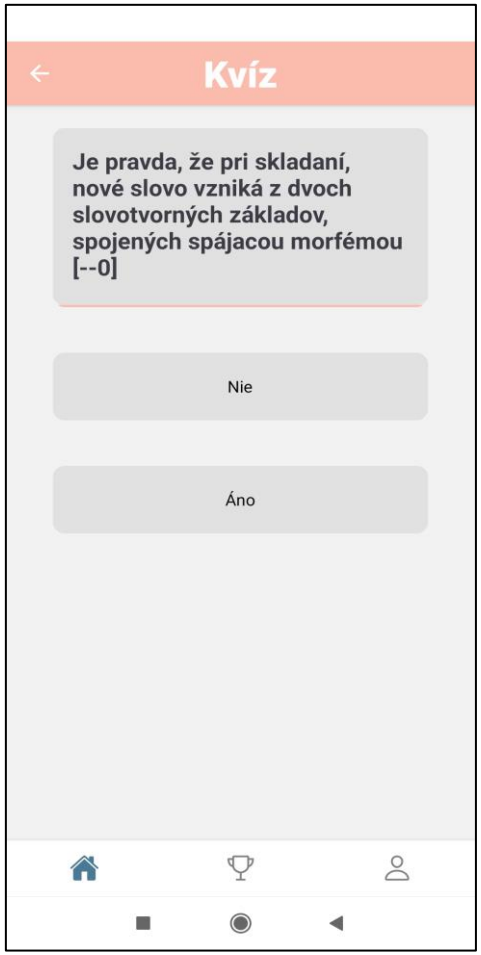
Rozdiely:

Najväčším rozdielom medzi aplikáciou **MATURANT** a **Maturita vo vrecku** je práve absencia maturitných testov – Maturita vo vrecku disponuje malým kvízovým systémom, kde si používateľ môže vybrať zameranie na dané otázky (Literatúra, gramatika, či zmiešané otázky z oboch) a otestovať sa na nich pomocou označovania odpovedí – aplikácia teda skôr slúži na zhrnutie potrebných učív a na otestovanie sa z nich. Oboje aplikácie podobne implementujú možnosť učiva k otázkam.



Obrázok 1 - Úvodná obrazovka aplikácie

Zdroj: [1]



Obrázok 2 – Kvízový systém aplikácie

Zdroj: [1]

Obrázok 3 – Sekcia „Gramatika“

Zdroj: [1]

1.2.2 Maturita

Aplikácia **Maturita** je podobnou aplikáciou ako aplikácia **Maturita vo vrecku**, avšak obsahuje témy na viaceré oblasti, ako je aj **Anglický jazyk**, **biológia**, **dejepis** a **mnoho ďalších**. Aplikácia však neobsahuje žiadne možnosti otestovania sa pomocou kvízu, keďže slúži len ako pomôcka pri učení sa a príprave, teda obsahuje len samotnú teóriu.

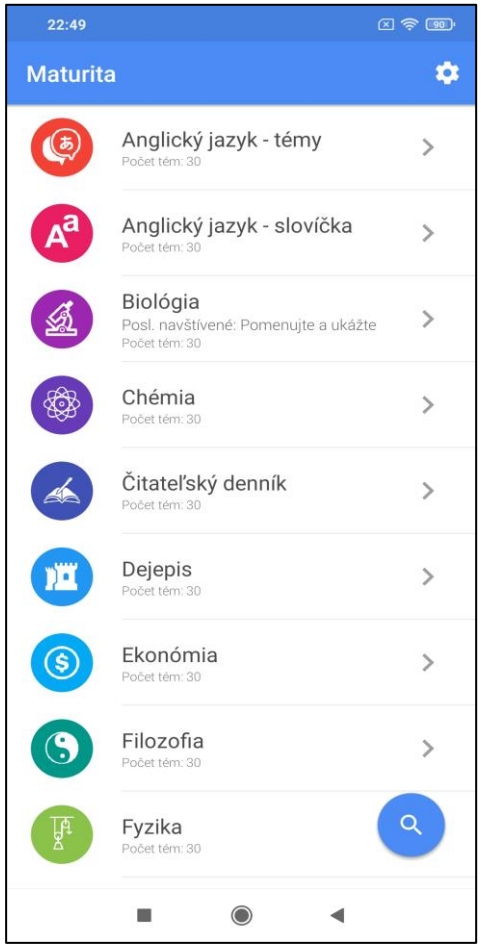
Funkcionality:

- Výber z viacerých okruhov, nie len Slovenského jazyka
- Možnosť rýchleho hľadania tém pomocou lupy
- Viacero nastavení prostredia – farba pozadia témy, farba textu témy, aplikácia na celú obrazovku, poslanie e-mailu autorovi

Aplikácia ma zaujala najmä tým že obsahovala témy na viaceré oblasti a teda ponúkala väčšiu multifunkčnosť, aj keď neobsahovala kvízový systém a ani nebola zameraná na maturitné testy.

Rozdiely:

Najväčším rozdielom medzi aplikáciou **Maturita** a **Maturita vo vrecku** je opäť absencia maturitných testov, či v tejto aplikácii dokonca forma hocijakého testu / kvízu. Aplikácia avšak pôsobí viac univerzálnejšie, keďže obsahuje viacero tém a nemá len jedno zameranie na slovenský jazyk.



Obrázok 4 - Úvodná obrazovka aplikácie

Zdroj: [2]

Obrázok 5 – Nastavenie pozadia témy aplikácie

Zdroj: [2]

Obrázok 6 – Témy Slovenského jazyka a literatúry

Zdroj: [2]

1.2.3 Autoškola

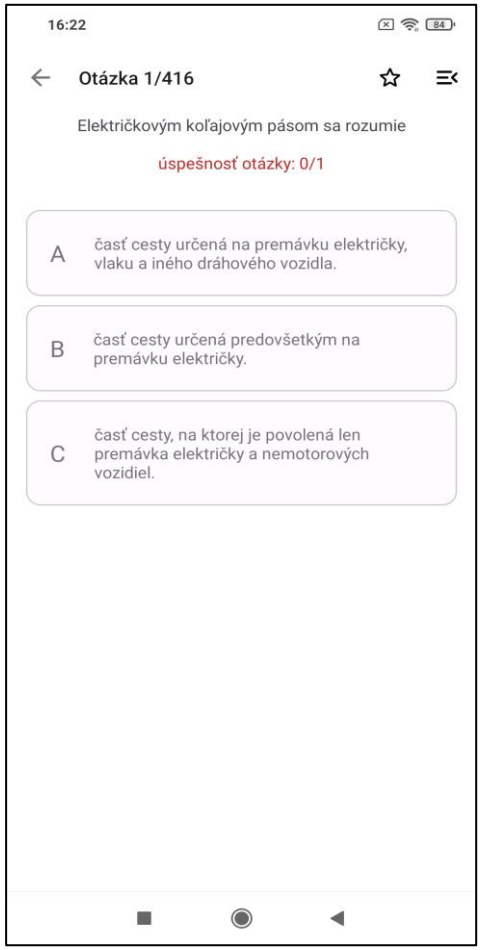
Aplikácia Autoškola je primárne iného zamerania – slúži na testovanie z otázok pravidiel cestnej dopravy, avšak aplikácia mi slúžila ako inšpirácia na nápad mojej aplikácie. Aj keď je aplikácia iného zamerania, jej podstata je rovnaká, keďže slúži na testovanie otázok z rôznych okruhov.

Funkcionality:

- Nastavenie jazyka aplikácie a samotných otázok na základe typu vodičského preukazu
- Vygenerovanie otázok z určitého okruhu
- Prístup k otázkam a správnym odpovediam mimo kvízu
- Implementované reklamy (aplikácia má aj platenú verziu bez obmedzení)
- Nastavenia obsahujúce štatistiky, možnosť vymazania dát
- Možnosť označiť otázky ako „obľúbené“
- Stiahnutie dostupných súborov

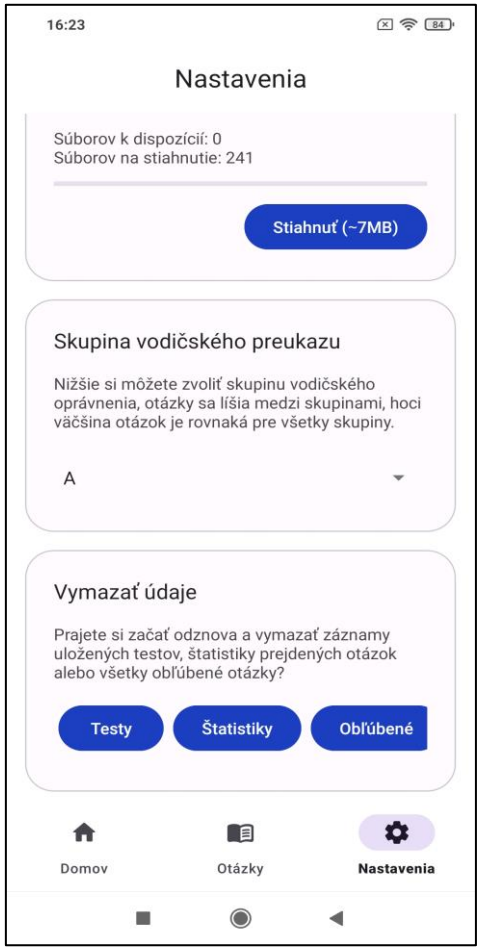
Rozdiely:

Aplikácia Autoškola disponuje viacerými nastaveniami – používateľ si môže vymazať všetky údaje (testy, štatistiky, obľúbené otázky), poskytuje možnosť si ukladať otázky medzi „obľúbené“ a má implementovaný sofistikovaný tester. Aplikácia je avšak iného zamerania a preto je samotná odlišnosť aplikácii vysoká.



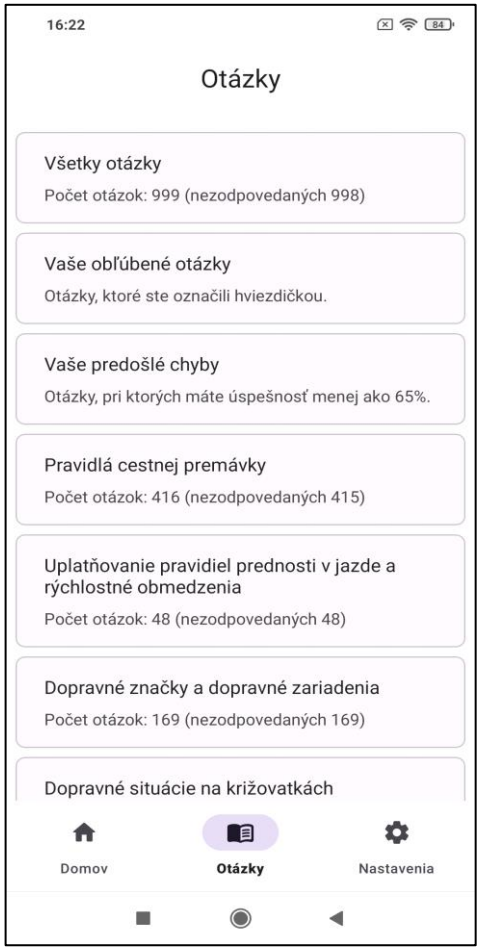
Obrázok 7 – Kvízový systém

Zdroj: [3]



Obrázok 8 – Nastavenia aplikácie

Zdroj: [3]



Obrázok 9 – Témy aplikácie

Zdroj: [3]

2 NÁVRH RIEŠENIA PROBLÉMU

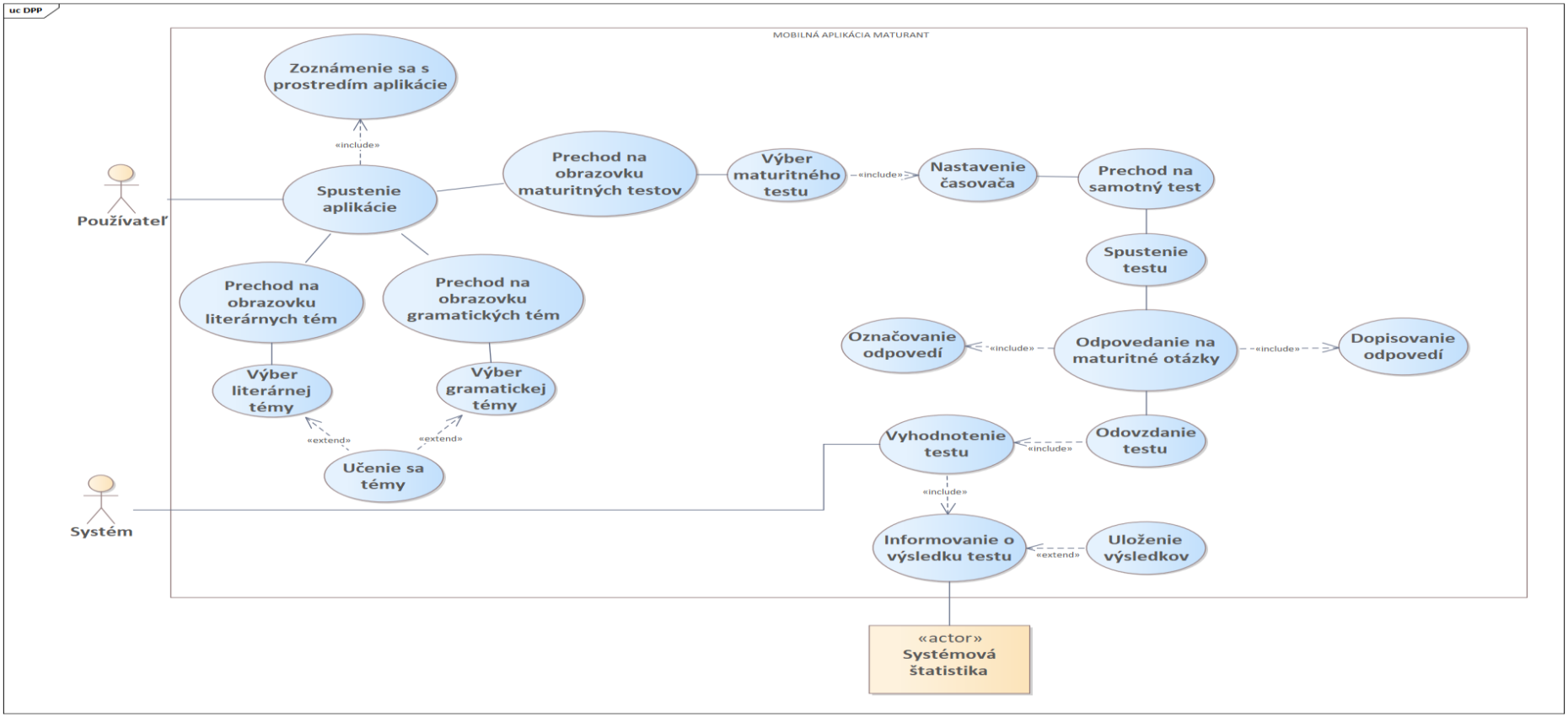
2.1 KRÁTKA ANALÝZA

Ako môžeme vidieť na diagrame prípadov použitia, aplikácia sa skladá z dvoch hlavných aktérov – používateľa a systému. Používateľ je najhlavnejší aktér ktorý ovláda aplikáciu, po spustení sa s aplikáciou oboznámi a následne má možnosť či chce prejsť na maturitné testy alebo má záujem si pozrieť gramatické alebo literárne témy. Po výbere maturitného testu si najprv musí používateľ nastaviť časovač, ktorý sa mu bude počas testu odpočítavať. Po nastavení časovača už prejde na samotný test, kde má možnosť odpovedať na otázky buď doplňovaním textu alebo označovaním odpovedí. Používateľ má hocikedy možnosť test odovzdať, čím sa test automaticky vyhodnotí (tu už pracuje druhý aktér – systém), pričom bude informovaný o výsledku svojich odpovedí a bude mať možnosť si uložiť výsledky testov, o čo sa opäť stará aktér Systém a sú uložené do aktéra Systémová štatistika.

Funkcionality:

- Prezeranie tém a učív zo slovenského jazyka a literatúry
- Tester na otázky z oficiálnych maturitných testov
- Nastavenie časovača pre daný test
- Ukladanie štatistík úspešnosti
- Vyhodnotenie testu

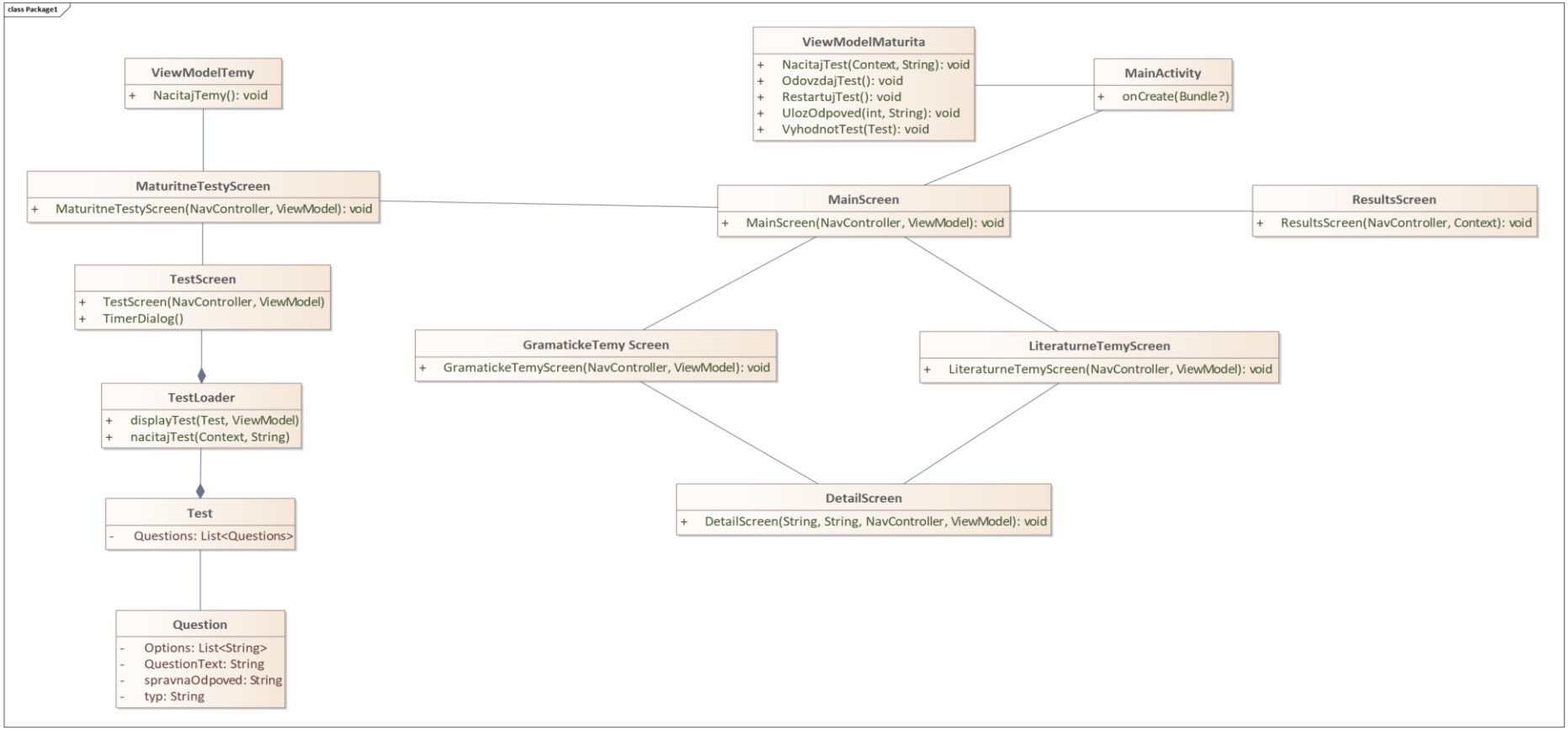
Čo sa týka všeobecnej analýzy, samotná aplikácia by mala slúžiť ako efektívny spôsob prípravy na maturitné skúšky. Aplikácia by mala byť podporovaná pravidelnou aktualizáciou obsahu, teda minimálne každým rokom keď sa objavia nové maturitné testy, či aktualizáciou starého obsahu za nový, poprípade pri zmene pravidiel samotného jazyka. Dôležitým bodom je taktiež dostupnosť aplikácie, mala by byť dostupná pre čo najviac používateľov ktorí používajú rôzne verzie Androidu. Samotný dizajn aplikácie by mal byť príjemný pre oči a prehľadný, aby mali používatelia možnosť sa sústrediť.



Obrázok 10 – Diagram prípadov použitia

Zdroj: Vlastné vypracovanie

2.2 NÁVRH APLIKÁCIE



Obrázok 11 – Diagram tried

Zdroj: Vlastné vypracovanie

3 POPIS IMPLEMENTÁCIE

3.1 MainActivity

Základnú časť aplikácie tvorí ***MainActivity***, ktorú som využil na vytvorenie navigácií pomocou knižnice ***NavHost*** a ***NavController***. Tým som mohol každej nezávislej ***Screen*** definovať navigáciu pomocou kliknutia. Taktiež som vytvoril pre niektoré triedy spoločný ***ViewModel*** s ktorým museli pracovať.

```
class MainActivity : ComponentActivity() {
    Mário <žilinčík>
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContent {
            MaturantTheme {
                val navController = rememberNavController()
                val viewModelMaturitaViewModel: MaturitaViewModel = viewModel()
                NavHost(navController = navController, startDestination = "MainScreen") { this: NavGraphBuilder
                    composable( route: "MainScreen") { MainScreen(navController) }
                    composable( route: "GrammarTopicsScreen") { GrammarTopicsScreen(navController) }
                    composable( route: "LiteratureTopicsScreen") { LiteratureTopicsScreen(navController) }
                    composable( route: "MaturitaTestScreen") { MaturitaTestScreen(navController, viewModelMaturitaViewModel) }
                    composable( route: "TestScreen") { this: AnimatedContentScope  it: NavBackStackEntry
```

3.2 MainScreen

MainScreen slúžila ako hlavná ***Screen*** na ktorej sa užívateľ objavil po zapnutí aplikácie. V ***scaffolde*** som aplikácii nastavil ***Info*** ikonu kde sa používateľ môže prekliknúť na ***Info Screen*** kde sú základné informácie o aplikácii. Cez ***content*** som ďalej nastavil obrázok ako pozadie a následne som v ***LazyColumn*** (definovaný v prípade že by používateľ otočil obrazovku) priradil ďalší obrázok ktorý slúži ako logo a cez *lambda* výraz pridal iteráciu ktorá vytvára ***@Composable*** funkcie - pridáva „odrážky“ ktoré slúžia na ďalšiu navigáciu v aplikácii.

```
fun MainScreen(navController: NavController, viewModel: SharedViewModel = viewModel()) {
    Scaffold(
        topBar = {
            TopAppBar(
                title = { Text( text: "") },
                actions = { this: RowScope
                    IconButton(onClick = { navController.navigate( route: "InfoScreen") }) {
                        Icon(Icons.Default.Info, contentDescription = "Informácie")
                    }
                }
            )
        },
        content = { innerPadding ->
            Box(
                modifier = Modifier
                    .fillMaxSize()
                    .padding(innerPadding)
                    .background(MaterialTheme.colorScheme.background)
            ) { this: BoxScope
```

```
item { Spacer(modifier = Modifier.height(32.dp)) }
// Lambda výraz zoberie všetky MenuItemInfo a v LazyItemScope cez ne iteruje a spracuje ich ako parameter pre MenuI
items(
    listOf(
        MenuItemInfo( text: "GRAMATICKÉ TÉMY", AppColors.UranianBlue, navigationDestination: "GrammarTopicsScreen"),
        MenuItemInfo( text: "LITERATÚRNE TÉMY", AppColors.SeaGreen, navigationDestination: "LiteratureTopicsScreen"),
        MenuItemInfo( text: "MATURITNÉ TESTY", AppColors.TuftsBlue, navigationDestination: "MaturitaTestScreen"),
        MenuItemInfo( text: "VÝSLEDKY", AppColors.Azul, navigationDestination: "ResultsScreen")
    )
) { this: LazyItemScope menuItem ->
    MenuItem(menuItem) {
        if (!viewModel.isNavigationLocked.value) {
            viewModel.lockNavigation()
            navController.navigate(menuItem.navigationDestination)
        }
```

3.3 GrammarTopicsScreen

Po kliknutí na gramatické témy sa dostaneme na obrazovku ***GrammarTopicsScreen***, ktorá obsahuje všetky gramatické témy. Parametrom je ***navController*** používaný na navigáciu a ***SharedViewModel***. Funkcia obsahuje ***LaunchedEffect*** vďaka ktorému sa automaticky načítajú štýly, ktoré používame (to prebieha vo ***ViewModeli***). Taktiež tu je implementovaný ***LazyColumn*** na scrollovanie ktorý si ukladá svoj stav vďaka ***rememberSaveable***. Po kliknutí na danú gramatickú tému sa opäť presmerujeme cez ***navController*** na ***DetailScreen***.

3.4 LiteratureTopicsScreen

Po kliknutí na literatúrne témy sa dostaneme na obrazovku ***LiteratureTopicsScreen,*** ktorá obsahuje všetkých literárnych autorov. Parametrom je opäť ***NavController*** používaný na navigáciu a ***SharedViewModel***. Funkcia obsahuje ***LaunchedEffect*** na automatické načítanie autorov (to prebieha vo ***ViewModeli***). Opäť tu je implementovaný ***LazyColumn*** ktorý si ukladá svoj stav vďaka ***rememberSaveable***. Po kliknutí na daného autora sa presmerujeme cez ***navController*** na ***DetailScreen***.

3.5 DetailScreen

Po kliknutí na určitú gramatickú tému alebo literárneho autora sa presmerujeme na ***DetailScreen*** ktorá už obsahuje samotné informácie o danej téme / autorovi. Na základe parametrov z predošlých tried vieme, či sa jedná **o gramatickú** alebo **literárnu** tému. Trieda opäť obsahuje ***LaunchedEffect*** na načítanie informácií o štýle / autorovi, pričom sa ***@Composable*** formátuje inak v závislosti od toho, či sa jedná o gramatickú alebo literárnu tému.

3.6 SharedViewModel

Tento ***ViewModel*** slúži na uchovávanie štýlov, autorov a informácii o tom, či sa dané štýly alebo autori práve načítavajú – pokiaľ áno, na ***DetailScreen*** sa ukáže ***CircularProgressIndicator*** (toto je implementované aj v ***LiteratureTopicsScreen*** a ***GrammarTopicsScreen***, aby sa predišlo „flashovaniu“ samotnej obrazovky a aby mal používateľ informáciu o tom, že sa dáta načítavajú.) Tento ***ViewModel*** taktiež obsahuje funkcie na načítanie štýlov a autorov z **.JSON** súboru a funkciu na zamknutie navigácie po prekliknutí sa na 1 sekundu. Pre načítanie dát z **.JSON** súboru som si vytvoril pomocné ***data classes*** – ***AuthorInfo*** a ***StyleInfo*** ktoré uchovávajú potrebné atribúty.

```kotlin
@Composable
fun GrammarTopicsScreen(navController: NavController, viewModel: SharedViewModel = viewModel()) {
    val context = LocalContext.current

    LaunchedEffect(Unit) { this: CoroutineScope
        viewModel.loadStyles(context)
    }


    val listState = rememberSaveable(saver = LazyListState.Saver) {
        LazyListState()
    }
```

```kotlin
fun loadStyles(context: Context) {
    if (_styles.value.isNotEmpty()) return
    _isLoadingStyles.value = true
    viewModelScope.launch { this: CoroutineScope
        val data = withContext(Dispatchers.IO) { this: CoroutineScope
            loadStylesFromJson(context)
        }
        delay( timeMillis: 250)
        _styles.value = data.associateBy { it.name }
        _isLoadingStyles.value = false
    }
}
```

Mário <žilinčík>

```kotlin
fun loadAuthorsFromJson(context: Context): List<AuthorInfo> {
    val inputStream = context.assets.open( fileName: "authors.json")
    val reader = InputStreamReader(inputStream)
    val type = object : TypeToken<List<AuthorInfo>>() {}.type
    return Gson().fromJson(reader, type)
}
```

3.7 MaturitaTestScreen

Po kliknutí na maturitné testy sa dostaneme na ***Screen*** ktorý zobrazuje všetky dostupné maturitné testy (označené podľa rokov). Po kliknutí na daný test sa používateľovi najprv ukáže ***TimerSetupDialog***, vďaka ktorému si nastaví časovač pre daný test. (Samotný časovač sa spustí v ***MaturitaViewModel***, ktorý si ukladá informácie o čase a aj samotný časovač.). Táto trieda veľmi spolupracuje s ***ViewModelom MaturitaViewModel*** – ukladá si informácie o čase, kliknutom roku, a to, kedy môže zobraziť dialóg.

3.8 TestLoader

TestLoader slúži ako pomocná trieda – obsahuje funkciu na načítanie testu z **.JSON** súboru. Opäť spolupracuje s **ViewModelom *MaturitaViewModel***, ktorý ukladá počet zodpovedaných otázok (potrebné napríklad pri odznačení odpovede, alebo pri vymazaní odpovede.). Samotná trieda sa stará o kompletné vyobrazenie testu pomocou sekcií v ktorých sú otázky a aj samotných obrázkov. Taktiež sa stará o vizuálne zmeny pri vyhodnotení testu, kliknutí na odpoveď či pri zodpovedaní otázky. Okrem toho obsahuje ďalšie pomocné ***@Composable*** funkcie – na „odovzdanie“ testu, legendu pri odpovediach, a pre ***TextField*** na odpovede používateľa.

3.9 TestScreen

TestScreen pomocou triedy ***TestLoader*** a ***MaturitaViewModel*** už zobrazuje načítaný test, pričom si všetky premenné získava práve z ***ViewModelu***. S pomocou ***ViewModelu*** pozorujeme najmä to, v akom stave sa práve ***Screen*** nachádza – **koľko ostáva času**, **či bol test vyresetovaný**, **či bol test odovzdaný**... Trieda obsahuje aj ďalšie ***@Composable*** funkcie na zobrazenie dialógu pri odchode z obrazovky či dialóg pri vypršaní času.

3.10 MaturitaViewModel

Tento ***ViewModel*** je kľúčový pre efektívne sledovanie stavov v ***TestScreen*** a ***MaturitaTestScreen***. Obsahuje všetky potrebné funkcie na **vyhodnotenie testu**, **načítanie testu**, **ukladanie otázok** (nutná implementácia – pri zmene orientácie obrazovky by sa inak všetky odpovede vymazali!), reštartovaní testu, pri kliknutí na daný rok či pomocné funkcie ako navýšenie a zníženie počtu odpovedí, pauznutie časovača...

```
Mário <žilinčík>
fun evaluateAnswers(test: Test) {
    var correctCount = 0
    var questionIndex = 0
    _questionResults.clear()

    test.sections.forEach { section ->
        section.questions.forEach { question ->
            val userAnswer = _userAnswers.getOrNull(questionIndex)?.trim()?.lowercase()
            val correctAnswers = question.correctAnswer.lowercase().split( ...delimiters: "|").map { it.trim() }

            val isCorrect = if (correctAnswers.contains("*")) {
                true
            } else {
                correctAnswers.contains(userAnswer)
            }

            _questionResults.add(isCorrect)
            if (isCorrect) correctCount++
            questionIndex++
        }
    }
    _testResults.value = Pair(correctCount, _userAnswers.size)
}
```

```
fun resetTimer() {
    viewModelScope.launch { this: CoroutineScope
        timerJob?.cancelAndJoin()
    }
}
```

```
fun startTimer(totalSeconds: Int) {
    _remainingTime.value = totalSeconds
    timerJob?.cancel()
    timerJob = viewModelScope.launch { this: CoroutineScope
        while (_remainingTime.value > 0) {
            delay( timeMillis: 1000)
            _remainingTime.value -= 1
        }
    }
}
```

3.11 ResultsScreen

ResultsScreen slúži ako ďalšia ***Screen*** ktorá umožňuje používateľovi zobrazenie jeho výsledkov pokiaľ sa rozhodol uložiť si výsledok nejakého testu. Na načítanie výsledkov používa systémovú zložku ***filesDir***, do ktorej ukladá údaje ***MaturitaViewModel***. Taktiež poskytuje možnosť výsledky vymazať.

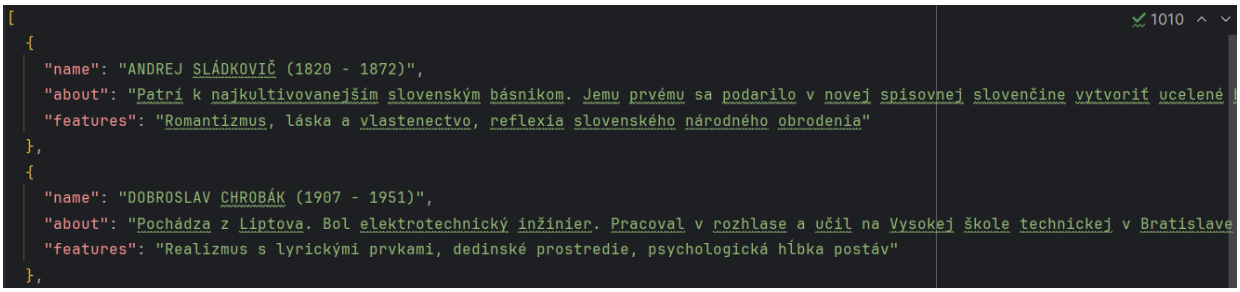
```
@Composable
fun ResultsScreen(navController: NavController, context: Context) {
    val resultsFile = File(context.filesDir, child: "test_results.txt")
    val results = remember { mutableStateListOf<String>() }
    var showDialog by remember { mutableStateOf( value: false) }

    LaunchedEffect(Unit) { this: CoroutineScope
        if (resultsFile.exists()) {
            resultsFile.forEachLine { line ->
                results.add(line)
            }
        }
    }
```

3.12 Načítavanie .JSON súborov

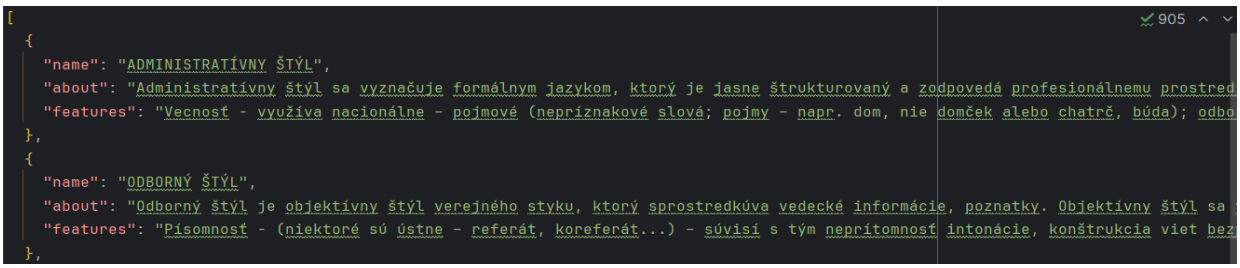
Na načítavanie a samotnú správu dát v **.JSON** súboroch som použil externú knižnicu ***GSON***. Vďaka nej som jednoducho mohol načítať údaje z **.JSON** súborov.

```
fun loadTestFromJson(context: Context, fileName: String): Test? {
    try {
        context.assets.open(fileName).use { inputStream ->
            val size = inputStream.available()
            val buffer = ByteArray(size)
            inputStream.read(buffer)
            val json = String(buffer, Charsets.UTF_8)
            val testContainer = Gson().fromJson(json, TestContainer::class.java)
            return testContainer.tests.firstOrNull()
        }
    } catch (e: Exception) {
        Log.e( tag: "LoadTest",  msg: "Error reading test file: $e")
        return null
    }
}
```

3.13 Štruktúra .JSON súborov

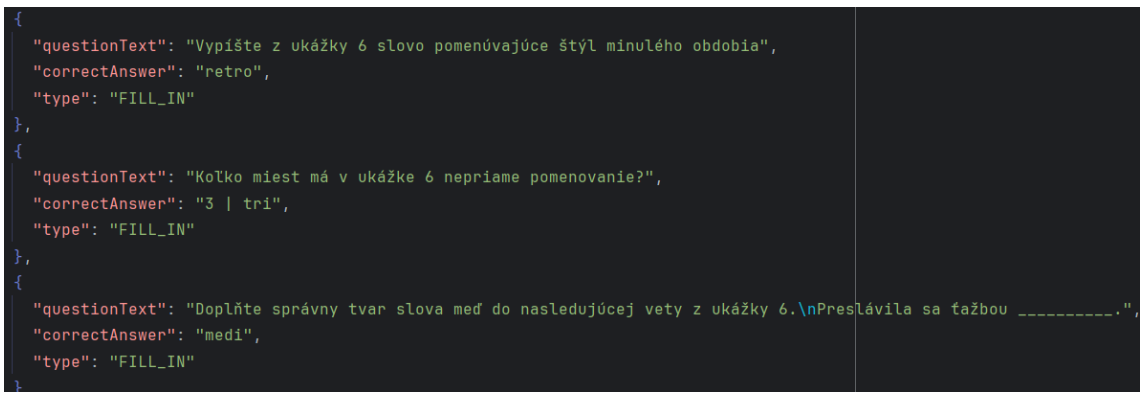


```
[
  {
    "name": "ANDREJ SLÁDKOVIČ (1820 - 1872)",
    "about": "Patrí k najkultivovanejším slovenským básnikom. Jemu prvému sa podarilo v novej spisovnej slovenčine vytvoriť ucelené
    "features": "Romantizmus, láska a vlastenectvo, reflexia slovenského národného obrodenia"
  },
  {
    "name": "DOBROSLAV CHROBÁK (1907 - 1951)",
    "about": "Pochádza z Liptova. Bol elektrotechnický inžinier. Pracoval v rozhlase a učil na Vysokej škole technickej v Bratislave
    "features": "Realizmus s lyrickými prvkami, dedinské prostredie, psychologická hĺbka postáv"
  },
```



```
[
  {
    "name": "ADMINISTRATÍVNY ŠTÝL",
    "about": "Administratívny štýl sa vyznačuje formálnym jazykom, ktorý je jasne štrukturovaný a zodpovedá profesionálnemu prostred
    "features": "Vecnosť - využíva nacionálne - pojmové (nepríznakové slová; pojmy - napr. dom, nie domček alebo chatrč, búda); odbo
  },
  {
    "name": "ODBORNÝ ŠTÝL",
    "about": "Odborný štýl je objektívny štýl verejného styku, ktorý sprostredkúva vedecké informácie, poznatky. Objektívny štýl sa
    "features": "Písomnosť - (niektoré sú ústne - referát, koreferát...) - súvisí s tým neprítomnosť intonácie, konštrukcia viet bez
  },
```

```
{
  "tests": [
    {
      "year": "2023",
      "sections": [
        {
          "sectionId": 1,
          "text": "Rozhovor so spisovateľom, prekladateľom, výtvarníkom a... Danielom Hevierom\nAlebo čuším, alebo kričím\n(úryvok)
          "questions": [
            {
              "questionText": "Ktoré tvrdenie vyplýva z ukážky 1?",
              "options": [
                "Ako spisovateľ prežíval Hevier kedysi tvorivú krízu.",
                "Hevier sa od útleho detstva túžil stať výtvarníkom.",
                "Hevierove aktivity zahŕňajú intenzívnu pedagogickú činnosť.",
                "Viaceré Hevierove knižky zostávajú aktuálne."
              ],
              "correctAnswer": "D)",
              "type": "CHOICE"
            },
            {
              "questionText": "Čo má na mysli Daniel Hevier pod vyjadrením ponorené knižky?",
              "options": [
                "Diela čakajúce na opätovné vydanie.",
                "Diela odsúdené na nedokončenie.",
                "Diela vyžadujúce čas na dotvorenie.",
                "Diela neurčené pre širokú verejnosť."
              ],
              "correctAnswer": "C)",
              "type": "CHOICE"
```



```
{
  "questionText": "Vypíšte z ukážky 6 slovo pomenúvajúce štýl minulého obdobia",
  "correctAnswer": "retro",
  "type": "FILL_IN"
},
{
  "questionText": "Koľko miest má v ukážke 6 nepriame pomenovanie?",
  "correctAnswer": "3 | tri",
  "type": "FILL_IN"
},
{
  "questionText": "Doplňte správny tvar slova meď do nasledujúcej vety z ukážky 6.\nPreslávila sa ťažbou _____.",
  "correctAnswer": "medi",
  "type": "FILL_IN"
}
```

4 ZOZNAM POUŽITEJ LITERATÚRY

[1] GOOGLE.COM. *Maturita vo vrecku*. [online] [cit 5.4.2024] Dostupné na: https://play.google.com/store/apps/details?id=sk.maturita.vo.vrecku

[2] GOOGLE.COM. *Maturita SK*. [online] [cit 5.4.2024] Dostupné na: https://play.google.com/store/apps/details?id=tursky.jan.maturita

[3] GOOGLE.COM. *Autoškola*. [online] [cit 5.4.2024] Dostupné na: https://play.google.com/store/apps/details?id=com.thematus.autoskola

[4] BAELDUNG.COM. *Converting Kotlin Data Class from JSON using GSON.* Dostupné na: https://www.baeldung.com/kotlin/json-convert-data-class

[5] GITHUB.COM. *GSON.* Dostupné na: https://github.com/google/gson

[6] BAELDUNG.COM *Using the Gson TypeToken in Kotlin.* Dostupné na: https://www.baeldung.com/kotlin/gson-typetoken

[7] ZMATURUJ.ZONES.SK. *Maturitné testy.* Dostupné na: https://zmaturuj.zones.sk/maturitne-testy/

[8] JAZYKOVEDKYNA.SK. Dostupné na: jazykovedkyna.sk

[9] DISCOVERIA.ORG. Dostupné na: discoveria.org

[10] REFERATY.AKTUALITY.SK. Dostupné na: referaty.aktuality.sk

[11] SLOVENCINA.VSELICO.COM. Dostupné na: slovencina.vselico.com